The PC Build Validator Challenge

1. Problem Statement

You've been hired as a junior developer for a PC building website. Your first task is to write a script that validates a list of "Featured Builds" for an upcoming promotion.

You will be given a **Component Inventory**, a list of **Build Kits**, and a fixed **Budget**.

Your script must process each Build Kit, check it against the Inventory for compatibility and price, and then determine which build offers the **highest performance score** while being both **compatible** and **under budget**.

2. Mechanics & Rules

2.1 Component Inventory

You are given a list of **P** components in your inventory. Each component has the following attributes:

Attribute	Description
component_id	Unique identifier (e.g., "cpu_1")
type	One of: CPU, Motherboard, GPU, RAM, PSU
performance_score	Integer score contribution
Cost	Price in dollars (integer)
Spec_1	First specification (see table below)
Spec_2	Second specification (see table below)

Specification Fields by Component Type:

Component Type	Spec_1	Spec_2
CPU	socket (e.g., "LGA1700")	TDP in watts (e.g., 95)
Motherboard	socket (e.g., "LGA1700")	RAM type (e.g., "DDR5")
GPU	NONE (always "-")	TDP in watts (e.g., 300)
RAM	RAM type (e.g., "DDR5")	NONE (always "-")
PSU	wattage (e.g., 750)	NONE (always "-")

Note: "-" represents no applicable specification for that field.

All numerical specs (performance_score, cost, wattage, TDP) are integers.

2.2 Build Kits

You are given **K** pre-defined build kits. Each kit contains exactly 5 component IDs:

- One CPU
- One Motherboard

- One GPU
- One RAM module
- One PSU

Format: kit_id cpu_id motherboard_id gpu_id ram_id psu_id

2.3 Budget

You have a single **Total_Budget** (e.g., $1500).

2.4 Goal

Find the kit_id that has the **highest total performance_score** (sum of all 5 components' scores) AND meets these two conditions:

- **Affordable**: sum(cost of all 5 parts) <= Total_Budget
- **Compatible**: The build must pass all validation checks (see below)

2.5 Compatibility Validation Rules

A build is **compatible** if and only if ALL of the following are true:

1. **CPU-Motherboard Socket Match**: cpu.socket == motherboard.socket
2. **RAM-Motherboard Type Match**: ram.ram_type == motherboard.ram_type
3. **PSU Wattage Sufficient**: psu.wattage >= (cpu.TDP + gpu.TDP **+ 50)** .The extra 50W accounts for other components (fans, storage, etc.)

Note: Socket comparison is **case-sensitive** (LGA1700 ≠ lga1700). RAM type comparison is **case-sensitive** (DDR5 ≠ ddr5)

2.6 Quick Algorithm Overview

Your solution should follow these steps:

1. Parse the budget and component inventory into a dictionary

2. For each build kit:

- Look up all 5 components
- Check if all components exist (skip if not)
- Calculate total cost and check budget
- Validate all 3 compatibility rules
- If valid, compare score with current best

3. Output the best build found (or NONE if no valid builds)

3. Input Format

B // Total Budget (integer)

P // Number of components in inventory (integer)

[P lines of components in format below]

K // Number of Build Kits to validate (integer)

[K lines of build kits in format below]

```

**Component Line Format:**

```
component_id type performance_score cost spec_1 spec_2
```

**Build Kit Line Format:**

```
kit_id cpu_id motherboard_id gpu_id ram_id psu_id
```

Example Input:

```
1500

8

cpu_1 CPU 500 300 LGA1700 95

cpu_2 CPU 450 250 AM5 105

mobo_1 Motherboard 150 180 LGA1700 DDR5

mobo_2 Motherboard 140 160 AM5 DDR4

gpu_1 GPU 700 400 - 300

gpu_2 GPU 600 350 - 250

ram_1 RAM 100 80 DDR5 -

psu_1 PSU 50 100 750 -

3

kit_A cpu_1 mobo_1 gpu_1 ram_1 psu_1

kit_B cpu_2 mobo_2 gpu_1 ram_1 psu_1

kit_C cpu_1 mobo_1 gpu_2 ram_1 psu_1

```

---

4. Output Format

```
Maximum Score: [highest_score]

Best Build: [kit_id]
```

Example Output:

```
Maximum Score: 1500

Best Build: kit_A
```

Explanation:

- kit_A:
  - Score: 500 + 150 + 700 + 100 + 50 = 1500
  - Cost: 300 + 180 + 400 + 80 + 100 = 1060✓ (under budget)
  - CPU socket (LGA1700) == Mobo socket (LGA1700) ✓
  - RAM type (DDR5) == Mobo RAM type (DDR5) ✓
  - PSU wattage (750) >= CPU TDP + GPU TDP + 50 = 95 + 300 + 50 = 445 ✓
- kit_B:
  - Cost: 250 + 160 + 400 + 80 + 100 = 990 ✓ (under budget)
  - Compatible sockets ✓
  - INCOMPATIBLE: RAM is DDR5 but mobo_2 requires DDR4 ✗
```

- kit_C:
  - Score: 500 + 150 + 600 + 100 + 50 = 1400
  - Cost: 300 + 180 + 350 + 80 + 100 = 1010 ✓ (under budget)
  - All compatibility checks pass ✓
  - But score (1400) < kit_A score (1500)

Winner: kit_A with maximum score of 1500.

---

5. Edge Cases & Special Rules

5.1 No Valid Builds

If all builds are invalid due to missing components, incompatibility, or exceeding budget, output:

```
Maximum Score: 0

Best Build: NONE
```

5.2 Ties

If multiple builds have the same maximum score, output the first one encountered in the input order.

5.3 Invalid Component References

If a build kit references a `component_id` that doesn't exist in the inventory, treat that build as invalid (skip it).

Example:

kit_X cpu_1 mobo_999 gpu_1 ram_1 psu_1

If mobo_999 doesn't exist in inventory, skip kit_X entirely.

5.4 All Components Must Exist

A build is only valid if it contains exactly 5 components (one of each type). If any component is missing from the inventory, the build is invalid.

---

6. Sample Test Cases

Test Case 1: Basic Valid Build

Input:

```
1000

5

cpu_1 CPU 400 200 LGA1700 65

mobo_1 Motherboard 100 150 LGA1700 DDR4

gpu_1 GPU 500 300 - 200

ram_1 RAM 80 60 DDR4 -

psu_1 PSU 40 80 650 -

1

kit_X cpu_1 mobo_1 gpu_1 ram_1 psu_1
```

Output:

```
Maximum Score: 1120

Best Build: kit_X
```

Test Case 2: No Valid Builds (All Incompatible)

Input:

```
2000
```

5

cpu_1 CPU 400 200 LGA1700 65

mobo_1 Motherboard 100 150 AM5 DDR4

gpu_1 GPU 500 300 - 200

ram_1 RAM 80 60 DDR4 -

psu_1 PSU 40 80 650 -

1

kit_Y cpu_1 mobo_1 gpu_1 ram_1 psu_1

```

Output:

```

Maximum Score: 0

Best Build: NONE

```

(Reason: CPU socket LGA1700 doesn't match Mobo socket AM5)

---

**Test Case 3: Over Budget**

Input:

```

500

5

cpu_1 CPU 400 200 LGA1700 65

mobo_1 Motherboard 100 150 LGA1700 DDR4

gpu_1 GPU 500 300 - 200

ram_1 RAM 80 60 DDR4 -

psu_1 PSU 40 80 650 -

1

kit_Z cpu_1 mobo_1 gpu_1 ram_1 psu_1

```

Output:

```

Maximum Score: 0

Best Build: NONE

7. Consider these questions as you develop your solution:

7.1 Object-Oriented Design

- How did you model components? Did you create a base Component class with subclasses for each type?
- Where did you implement the is_compatible() logic? In a PCBuild class?

Hint: Create a base class Component, and subclasses like CPU, Motherboard, etc. Each should have attributes and possibly a get_spec() method. Then a PCBuild class can handle compatibility checking.

7.2 Data Structures

- What's the most efficient way to store the inventory for quick lookups by component_id?
  - *Hint: Dictionary/HashMap with O(1) lookup is ideal*
- How did you store and iterate through build kits?

7.3 Algorithm Design

- What's the time complexity of your solution?
  - *Expected: O(K) for K build kits, with O(1) component lookups*
- How do you track the "best build seen so far"?

7.4 Error Handling

- How do you handle missing components gracefully?
- What if a PSU wattage or TDP value is invalid (negative, non-numeric)?

8. Input Handling

- Your program should read input from standard input (e.g., using input() in Python). You may assume that all fields are space-separated